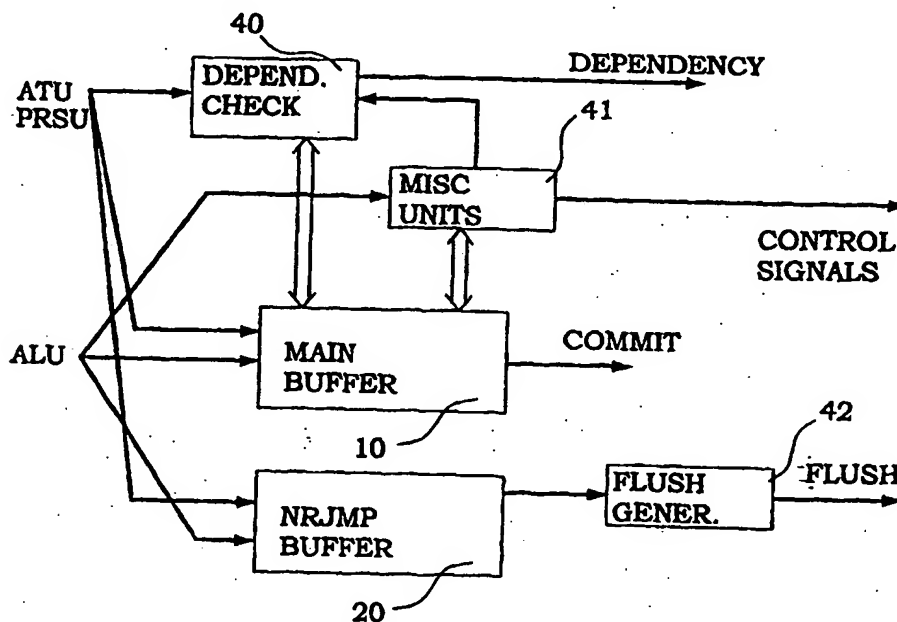




INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁷ : G06F 9/38		A1	(11) International Publication Number: WO 00/42500
			(43) International Publication Date: 20 July 2000 (20.07.00)
(21) International Application Number: PCT/SE99/02361 (22) International Filing Date: 15 December 1999 (15.12.99) (30) Priority Data: 9900042-4 11 January 1999 (11.01.99) SE (71) Applicant: TELEFONAKTIEBOLAGET LM ERICSSON [SE/SE]; S-126 25 Stockholm (SE). (72) Inventors: BERGLIN, Pär, David; Lundavägen 57C, S-212 24 Malmö (SE). PETTERSSON, Hans, Christian; Håkan Berg, Siriusgatan 32, S-415 22 Göteborg (SE). (74) Agents: STENBORG, Anders et al.; Aros Patent AB, Box 1544, S-751 45 Uppsala (SE).		(81) Designated States: AE, AL, AM, AT, AT (Utility model), AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, CZ (Utility model), DE, DE (Utility model), DK, DK (Utility model), DM, EE, EE (Utility model), ES, FI, FI (Utility model), GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KR (Utility model), KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG). Published With international search report. Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.	

(54) Title: NOT REPORTED JUMP BUFFER



(57) Abstract

The present invention achieves a fast handling of the predicted jumps by introducing an additional buffer (20) for not reported predicted jump instructions in a reorder buffer. This additional buffer (20) is separate from the main buffer (10) and is supplied only with information associated with instructions related to predicted jumps. A predicted jump instruction is preferably stored both in the main buffer (10) and the additional buffer (20). The additional buffer operates in parallel with the main buffer (10) and is designed as a linear first-in-first-out queue. The first not reported jump is then always easily available at the top of the queue for evaluating the jump conditions. If a mispredicted jump is determined, the reorder buffer is flushed by a flush generator unit (42).

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon	KR	Republic of Korea	PL	Poland		
CN	China	KZ	Kazakhstan	PT	Portugal		
CU	Cuba	LC	Saint Lucia	RO	Romania		
CZ	Czech Republic	LI	Liechtenstein	RU	Russian Federation		
DE	Germany	LK	Sri Lanka	SD	Sudan		
DK	Denmark	LR	Liberia	SE	Sweden		
EE	Estonia			SG	Singapore		

NOT REPORTED JUMP BUFFER

TECHNICAL FIELD

5 The present invention relates to the field of processors and in particular to reorder buffers in superscalar processors using out-of-order execution.

TECHNICAL BACKGROUND

10 In order to increase the capability of modern processors, the development is directed to solutions, where the processors execute several tasks in parallel. A superscalar processor handles more than one instruction each cycle. The reason for this development is that the speed of accessing memories has not increased in the same rate as the increased processor speed. With
15 parallelism, more accesses can be done and even if the delay of each access is the same, more accesses can still be done over the same period of time.

20 The handling of large number of concurrently executed instructions has large benefits. Some of the benefits are pronounced when also using out-of-order execution, which nowadays often is employed. If an instruction has to wait for its operands a long time other instructions behind, which already have its operands ready can be executed while the first instruction waits for operands. This reduces the impact of a slow memory and maintains a higher throughput of instructions.

25 When using out-of-order execution, different hazards may occur. In such cases, a reorder buffer is introduced to keep track on the instructions, their results, their internal dependencies and order them in a proper program sequence.

30 In a superscalar processor according to prior art, predicted jumps are often used. A jump instruction may be conditioned on a certain result of another instruction, and the definite jump will not be possible to perform if the result

is not available. However, if the jump instructions have to wait for the previous result to be ready for a long time, the overall speed of the processor is reduced. In order to speed up the process, the processor makes a prediction of whether the jump is going to be taken or not, and the subsequent instructions may be executed based on this. If the predicted jump was found to be correct at a later occasion, when the real result upon which the jump decision has to be made is available, the process may be continued from the point it had reached in the meantime. If the predicted jump was mispredicted, the processor has to start all over from the jump instruction and take the other path after the mispredicted jump. This means that all the instructions that are stored in the reorder buffer and other modules following the mispredicted jump instruction, have to be flushed. The probability for a correct prediction is much larger than a misprediction, so that the gain in time and reduced complexity by letting the execution continue past a predicted jump is larger than the loss caused by a flush and related actions.

In reorder buffers according to the state of the art, the processor stores all instructions in a main buffer together with associated information, e.g. concerning reading and writing addresses, sequence number etc. The main reorder buffer is normally arranged as a first-in-first-out queue in order to keep the original sequence order. The instructions are executed, and the results are temporarily stored in the reorder buffer. When the instructions have reached the first position of the buffer, i.e. when all earlier instructions are removed, a check, that the instruction is executed and that all results are normal and can be written into intended storage, is performed, the results are subsequently written into appropriate positions and the instruction is removed from the reorder queue. This procedure is known as committing the instruction.

Since a mispredicted jump instruction may cause an extensive loss of executed instructions, the evaluation of predicted jumps is requested to be performed as quickly as possible. The information stored together with the

first not reported predicted jump has to be extracted from the main buffer. This information is then evaluated by the jump result that is received from the Arithmetic Logic Unit (ALU) and reported. If the prediction was correct, a flag is set in the main buffer that the predicted jump is reported to be correct and the processor searches for the next "not reported jump". On the contrary, if the prediction was wrong, a flush signal is sent to the processor. The processor favours the evaluation of the jumps but has the disadvantage of introducing a search procedure in the main buffer. When designing processors, the problem with this solution is that the extraction of information takes too long time, in particular with a large reorder buffer and when the processor runs at a high clock speed. This means that the mispredicted jumps are then followed by too many executed instructions before they are evaluated. Since the evolution of processors tends to increase the clock speed of the processor as well as the size of the reorder buffer, there is a general problem to reduce the time and efforts for predicted jump evaluation.

DESCRIPTION OF THE INVENTION

It is a general object of the present invention to reduce the time used for evaluation of predicted jumps in a reorder buffer. It is also an object of the present invention to achieve such time reductions with as limited additional hardware means as possible, preferably less.

The above objects have been achieved by a device and method according to the accompanying claims.

In general words, the present invention has solved the problem to achieve a fast handling of the predicted jumps by introducing a separate buffer for not reported predicted jump instructions in the reorder buffer. This buffer is separate from the main buffer and contains only instructions related to predicted jumps. A predicted jump instruction is preferably stored both in the main buffer and the additional buffer. This "not reported jump queue"

operates in parallel with the main buffer and is designed as a linear first-in-first-out queue. The first not reported jump is then always easily available at the top of the queue and may be evaluated and reported very rapidly.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is described in more detail below in connection with the enclosed drawings, in which:

- Fig. 1 is a schematic illustration of the distribution of instruction information according to the present invention;
- Fig. 2 is a block diagram showing an instruction processor unit, in which a reorder buffer according to the invention can be used;
- Fig. 3 is a block diagram showing an instruction queue controller, in which a reorder buffer according to the invention can be used;
- Fig. 4 is a block diagram of a reorder buffer according to the prior art;
- Fig. 5 is a block diagram of a first embodiment of a reorder buffer according to the present invention;
- Fig. 6 is a flow diagram, illustrating the process of storing instructions in a reorder buffer according to the present invention;
- Fig. 7 is a flow diagram, illustrating the process of storing results and evaluating predicted jump conditions according to the present invention; and
- Fig. 8 is a flow diagram, illustrating the process of committing instructions from a reorder buffer.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

In fig. 1, the instruction information flow in a reorder buffer according to the present invention is illustrated. A reorder buffer according to the present invention comprises a main buffer 10, in which instructions and associated information is stored. The instructions are identified and ordered by a tag, hereinafter referred to as the sequence number. The reorder buffer according

to the present invention also comprises an additional buffer 20, hereinafter referred to as a "not reported jump buffer", in which the sequence numbers of predicted jump instructions are stored, preferably together with selected associated information. The main buffer 10 contains a number of positions 18, in which information about instructions are to be placed. One buffer position 18 is intended for one instruction. Each buffer position 18 has a number of fields 11-16 in which information associated to the instruction can be stored. The not reported jump buffer 20 comprises in a similar manner a number of jump buffer positions 19, which preferably is less than the number of positions 18 in the main buffer. Each jump buffer position has a number of fields, a part of which may be a subset of the fields of the main buffer positions. Both buffers are normally designed as first-in-first-out (FIFO) queues, which means that the entries of information determines the order in which the information leaves the buffers.

Information 30 concerning a new instruction is sent to the reorder buffer. The information normally comprises the actual instruction 11 or a tag to or another representation of the instruction, and the sequence number 12, which indicates the order in which the instructions 11 are to be committed. Associated information may also comprise fields indicating data sources 13, i.e. which variables or memory contents that should be used as operands of the instruction, result destination 14, i.e. where the result from the instruction is going to be stored, and other important information. For a predicted jump instruction, the predicted condition for the jump may be included, either as a part of the instruction or as separate information. If the jump is of the form of "Branch if not equal to zero", the condition for the jump is obvious from the instruction itself and no further information has to be stored. The information 30 about the new instruction is added to the first available position in the main buffer 10 at the end of the FIFO queue. If the new instruction is a predicted jump, information is also supplied to the not reported jump buffer 20. Preferably, the sequence number 11 and information about the predicted condition for the jump 17 is stored at the first available position in the not reported jump buffer 20. Since all evaluation of predicted

jumps are going to be performed from the not reported jump buffer, the information about the predicted jumps may be minimised in the main buffer, and thereby reduce the necessary available space.

5 In order to evaluate the first not reported predicted jump instruction, which is hidden somewhere in the main buffer 10, the first position in the not reported jump buffer 20 is selected. This first jump instruction is evaluated and reported. In the present invention, no report flag has to be set in the main buffer 10, the instruction just has to be removed from the not reported
10 jump buffer 20. The correctly predicted jump instruction is thus removed from the not reported jump buffer 20 and the not reported jump buffer 20 is popped, i.e. all positions are moved forward one step. When the not reported jump buffer 20 is popped, the next instruction in the not reported jump buffer 20 is ready to be evaluated. If a misprediction is found, a flush signal
15 has to be sent as soon as possible, removing all instructions, with a higher sequence number than the mispredicted jump, from the main buffer 10 as well as the not reported jump buffer 20.

20 The time for evaluation and extraction of crucial information reduces considerably using such a separate queue for predicted jump instructions, since the first instruction in the queue always is the next one to be treated, and no searching in a large buffer has to be performed.

25 This proposed method according to the present invention may involve the creation of an extra copy of some information, since it may appear both in the main buffer 10 and in the not reported jump buffer 20, even if the amount of the copied information is limited. However, if the number of positions and fields in the not reported jump buffer 20 is kept low, this extra storage only takes a small amount of extra area. This extra area used is
30 often negligible in comparison with the logic needed for searching the main buffer 10 for information and the space omitted in the main buffer 10. In certain processors, the space reduction of the main buffer 10, arising from

the possibility to omit some predicted jump information, may alone be larger than the space required by the additional not reported jump buffer.

5 The present inventions thus provides a method which performs predicted jump evaluation and reports very quickly, which implies that any execution of instructions after a mispredicted jump is reduced, and the over-all process time is shortened. The method is generally applicable to all processors using jump prediction.

10 In the following, a presently preferred embodiment of the present invention will be described in some detail. This particular described processor constitutes a part of a telecommunication system, but the reorder buffer according to the present invention is useful in many types of processors using predicted jumps.

15 In a central processing unit (CPU) of the processor system there are basically two large components, a signal processor unit (SPU) and an instruction processor unit (IPU). The SPU handles job level execution, in which order jobs are to be executed, and the IPU executes the programs of each job. The
20 reorder buffer is situated in the IPU, why only the IPU will be further described. First an overview of a processor design is given, in order to define the role of the reorder buffer.

25 Fig. 2 illustrates a schematic block diagram of an IPU in a processor 45. The processor 45 also comprises SPU (not shown) and other auxilliary units (not shown) according to the knowledge of anyone skilled in the art. However, these parts are not central for the understanding of the present invention and any conventional technology may be used. An instruction queue controller IQC 53 communicates with a SPU via a signal processor interface
30 SPI 55. The IPU receives jobs from the SPU. The job decides what programs are going to be executed in the IPU. A program reference store unit PRSU 50 keeps track of the programs in assembler code ASA. When a program is going to be executed it is fetched from a program reference store PRS 51

5 together with prediction information for conditional jumps. The ASA instructions and the predictions are sent to an address translation unit ATU 52 for decoding. The ATU 52 translates assembler instructions into micro instructions MIP. The instructions are decoded and if an instruction is a conditional jump, the jump is predicted with help from the PRSU 50. The decoded micro instructions are then sent to the IQC 53.

10 A data store DS 57 is the memory for data in the IPU and contains all the information necessary. A data store handler DSH 56 handles the accesses to the DS 57. A register memory unit RMU 54 contains register memory and link register stacks. Registers are written and read by instructions. Also other units may be included in the IPU, but since they do not affect the operation of the reorder buffer, the description of such detail parts will be excluded.

15 Fig. 3 illustrates a simple design of a IQC. Instructions from the PRSU and the ATU are together with a unique sequence number sent to reservation stations RSV 60 and to a reorder buffer ROB 62. The RSV 60 holds the instruction until all dependencies of other results have been solved and the
20 operands have been fetched from memory and registers. It has normally several different queues to store the different types of instructions. When an instruction has all the operands, it is sent through an arithmetic logic unit ALU 61. Operands can be received from several different sources, e.g. DSH, RMU or hardware registers and status registers HR inside the IQC.

25 The ALU 61 executes all different types of instructions and stores the result in the reorder buffer ROB 62, normally within one clock cycle. If an instruction in the RSV 60 needs the result from another instruction it can receive the result from the ALU 61 or directly from the reorder buffer if the
30 result is already calculated.

The reorder buffer ROB 62 stores all instructions received from the ATU 61. When an instruction is calculated, the result is stored in the reorder buffer

together with the associated instruction. When the instructions are ready they are then committed, in the same order as they were received by the reorder buffer. The reorder buffer also keeps track of the predicted jumps, and flushes the IPU if mispredicted jumps are occurring. The committing involves writing of the results in different manners, e.g. to the RMU or DSH. Results may also be sent to the SPI.

In order to better understand the important advantages with reorder buffers according to the present invention, a description of a reorder buffer according to the prior art is included. In fig. 4, a typical design of a reorder buffer according to prior art is disclosed. Instructions from the ATU or PRSU are received and stored in a main buffer 10, normally constituted as a first-in-first-out buffer. New instructions are inserted last into the buffer in the first empty positions, and instructions are committed from the first position. The instructions are also sent to a dependency check unit 40 for comparing the destination operands with operands of new incoming instructions. If there is a dependency, necessary information is sent to the RSV.

Results from the ALU are inserted into correct positions in the main buffer 10. A unit for miscellaneous functions 41 also receives the results from the ALU and in cooperation with the main buffer 10, it keeps track on different maintenance and control functions. The miscellaneous unit 41 is e.g. responsible for searching the main buffer for the first not reported jump instruction and the evaluation of the predicted condition, which in cases of misprediction will lead to a flush of the reorder buffer, by a flush generator unit 42. The miscellaneous unit 41 also supports the dependency check unit 40 with appropriate information.

In fig. 5, a reorder buffer according to a preferred embodiment of the present invention is illustrated. Instructions from the ATU or PRSU are received and stored in a main buffer 10, preferably constituted as a first-in-first-out buffer. New instructions are inserted last into the buffer in the first empty positions, and instructions are committed from the first position. The

instructions are also sent to a dependency check unit 40 for comparing the destination operands with operands of new incoming instructions. If there is a dependency, necessary information is sent to the RSV. If the instruction from the ATU or PRSU is a predicted jump instruction, at least the sequence number and preferably also information of the condition for the jump instruction is also sent to a not reported jump buffer 20. The not reported jump buffer is preferably designed as a first-in-first-out buffer, and the new instruction is inserted into the first empty position of the buffer 20. Since only predicted jump instructions are inserted, the first position in the not reported jump buffer 20 is always occupied by the first not reported jump.

Results from the ALU are inserted into correct positions in the main buffer 10. A unit for miscellaneous functions 41 also receives the results from the ALU and in cooperation with the main buffer, it keeps track on different maintenance and control functions. The miscellaneous unit 41 also supports the dependency check unit 40 with appropriate information. The not reported jump buffer 20 is also supplied with appropriate results and is responsible for finding the first not reported jump instruction. Since this always corresponds to the instruction associated with the first position, the search is trivial and therefore extremely fast. The predicted jump condition is evaluated if the result for the predicted jump in the first position in the not reported jump buffer is available. In the case of a correct prediction, the first position of the not reported jump buffer 20 is removed and the buffer is popped. In cases of misprediction, a flush generator 42 is used to flush the reorder buffer from instructions having a sequence number larger than the jump instruction giving rise to the flush.

It would be possible to store the predicted jump instructions only in the not reported jump buffer 20, but the reporting of correct predictions will then lead to the need of incorporating the jump instruction into the middle of the main buffer 10. More preferable is thus to store only necessary information in the not reported jump buffer 20. This may leads to a double storage of certain information, if the information is a copy of the information available

in the main buffer, but it is negligible in comparison with the benefit of the arrangement. As an example, assume that the main buffer 10 contains 64 positions and 5 registers of each position determining conditions of the jump, which are unnecessary if a separate buffer is used. The introduction of a separate register will thus save 320 registers. If the not reported jump buffer 20 has 10 positions occupying 50 registers each, 500 registers have to be used, i.e. 180 registers more. However, when taking into account that 50 OR-gates and 50 AND-gates per main buffer position are saved by excluding the extraction of information from the main buffer 10, a total of 3200 OR-gates and 3200 AND-gates are saved by the present invention. Furthermore, for other processors, the size of the removal of registers from the main buffer may even exceed the size of the total not reported jump buffer.

The procedure for handling predicted jump instructions in a reorder buffer may be viewed as three separate part processes, dealing with storing of instructions and results, checking predictions and committing instructions. In fig. 6, the part process of inserting instructions in the reorder buffer is illustrated. The part process starts in step 100. In step 102 a new instruction from the ATU or PRSU is received in the reorder buffer. The instruction is in step 104 stored in the main buffer. In step 106, it is decided whether or not the instruction is a predicted jump instruction. If the instruction is a predicted jump instruction, information associated with the instruction is stored in the not reported jump buffer in step 108. The information may at least to a part be a copy of information in the main buffer. The part process stops in 110.

In Fig 7, the part process of handling results and in particular the evaluation of the predicted jump instructions, is illustrated. This part-process starts for each clock cycle in step 112. In step 113 a new result is received from the ALU. The result is stored together with the associated instruction in the main buffer in step 114. In step 115, it is decided whether or not the result was associated with a predicted jump instruction. If the result was a result of a predicted jump, the result is also stored in the not reported jump buffer

in step 116. If the result was not a result of a predicted jump, the part process continues directly to step 117. In step 117 the first position of the not reported jump buffer is selected, which corresponds to the first not reported predicted jump instruction. In step 118 it is checked if the results determining the real condition of the jump is available. If they are not available, the part process is ended at 122. The validity of the prediction of the jump can then not yet be determined. If appropriate results are available, a prediction evaluation is performed in step 119. If the prediction was correct, i.e. a correct jump has been performed, the first position in the not reported jump buffer is removed in step 120 and all subsequent positions are shifted up one step, i.e. "popped". The part process then ends at 122. If a misprediction was made, i.e. the assumed jump was incorrect, a flush of the reorder buffer for instructions with higher sequence numbers is performed in step 121 together with the reporting of the present predicted jump. Since no further predicted jump instructions then are available, the part process ends in step 122.

The commit part process is not directly affected by the present invention, but for completeness of the description, it is illustrated in Fig. 8. The part process starts in step 126. In step 128, the process waits for the instruction of the first position of the main buffer to be ready to be committed. In step 130, the results of the instruction is written into appropriate memories, and in step 132, the instruction is removed from the main buffer and the subsequent instructions are shifted one step forward. The part process ends in step 134.

From the above description, the benefits of the present invention are easily understood. By extracting predicted jump related information from the main buffer already at the insertion phase, the subsequent evaluation of the predicted jumps are facilitated. The extra space needed for the not reported jump buffer is, at least to a part, compensated by a reduced need for information in the main buffer. Furthermore, additional evaluation means in the main buffer may be reduced significantly. The actual procedure of

evaluation is possible to perform more rapidly, since the step of finding the first not reported predicted jump becomes trivial. This, in turn, leads to a faster overall evaluation time.

- 5 It will be understood by those skilled in the art that various modifications and changes may be made to the described embodiments of the present invention without departing from the scope thereof, which is defined by the appended claims.

CLAIMS

1. A reorder buffer (62) comprising a main buffer (10) arranged for storage of a plurality of instructions (11) and associated information (12-16),
5 **characterised in that**
said reorder buffer (62) further comprises an additional buffer (20) arranged for storage of information associated with predicted jump instructions (12, 17).

10 2. The reorder buffer (62) according to claim 1, **characterised in that** at least a part of the content of said additional buffer (20) also is stored in said main buffer (10).

15 3. The reorder buffer (62) according to claim 1 or 2, **characterised in that** said additional buffer (20) is arranged as a first-in-first-out queue.

20 4. The reorder buffer (62) according to any of the preceding claims, **characterised in that** said additional buffer (20) has a number of positions (19) which is considerably less than the number of positions (18) of said main buffer (10).

25 5. The reorder buffer (62) according to claim 4, **characterised in that** each of said positions (19) of said additional buffer (20) has a number of fields which is less than the number of fields in each of said positions (18) of said main buffer (10).

30 6. The reorder buffer (62) according to any of the preceding claims, **characterised in that** each position (19) of said additional buffer (20) at least comprises a sequence number (12) associated with a predicted jump instruction.

7. The reorder buffer (62) according to claim 6, **characterised in that** each position (19) of said additional buffer (20) also comprises information of the prediction (17) used by said predicted jump instruction.

5 8. The reorder buffer (62) according to any of the preceding claims, **characterised in that** an output terminal of said additional buffer (20) is connected to a flush generator unit (42).

10 9. A superscalar processor having a reorder buffer (62) according to any of the preceding claims.

10. A method for handling of predicted jumps in a reorder buffer (62) comprising the steps of:

15 storing instructions and associated information in a main buffer (10) of said reorder buffer (62), said instructions including predicted jump instructions;

storing results of executed instructions in said main buffer (10);

finding the oldest not reported predicted jump instruction;

20 if said results agrees with the predictions of said oldest not reported jump instruction, reporting said oldest not reported jump instruction as correct; and

if said results disagrees with the predictions of said oldest not reported jump instruction, flushing all instructions in said reorder buffer (62) entered after said oldest not reported jump instruction, and

25 **characterised in that**

said method comprises the further step of:

storing information associated with predicted jump instructions in an additional buffer (20) in said reorder buffer (62); and

30 storing results associated with predicted jump instructions in said additional buffer (20);

whereby the step of finding the oldest not reported predicted jump instruction is performed on said additional buffer (20).

11. The method for handling of predicted jumps according to claim 10, **characterised by** the step of storing a copy of at least a part of the information associated a predicted jump instruction stored in said main buffer (10) of said reorder buffer (62), in said additional buffer (20).

5

12. The method for handling of predicted jumps according to claim 10 or 11, **characterised by** the step of storing a sequence number (12) associated with a predicted jump instruction in said additional buffer (20).

10

13. The method for handling of predicted jumps according to claim 10, 11 or 12, **characterised by** the step of storing information of the prediction (17) used by said predicted jump instruction in said additional buffer (20).

15

14. The method for handling of predicted jumps according to any of the claims 10 to 13, **characterised by** storing information in said additional buffer (20) in a first-in-first-out manner, whereby said step of finding the oldest not reported predicted jump instruction comprises the selection of the first position in said additional buffer (20).

20

15. The method for handling of predicted jumps according to any of the claims 10 or 14, **characterised in that** said step of reporting said oldest not reported jump instruction as correct comprises the deletion of the first position in said additional buffer (20).

25

1/7

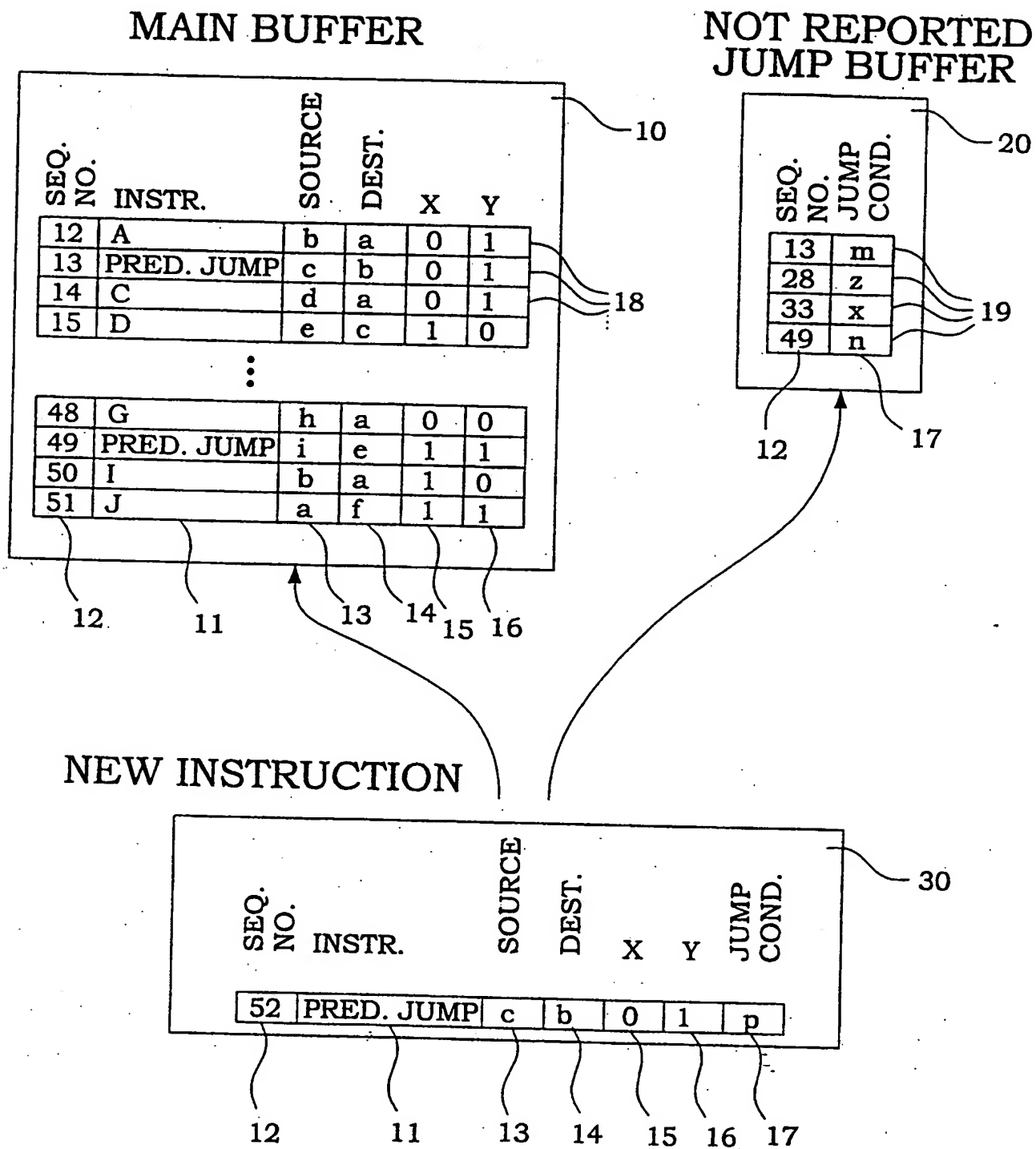


Fig. 1

2/7

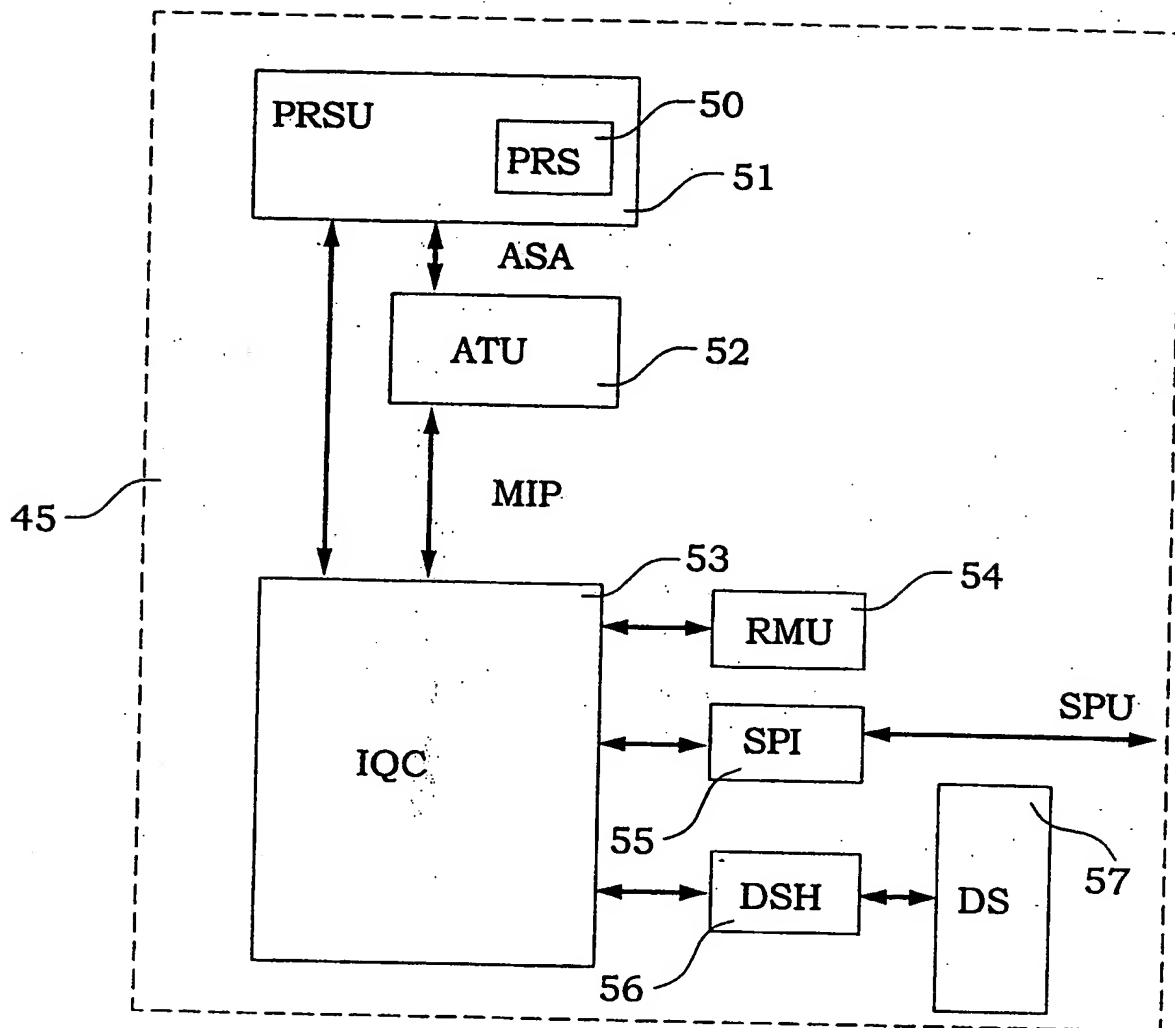


Fig. 2

3/7

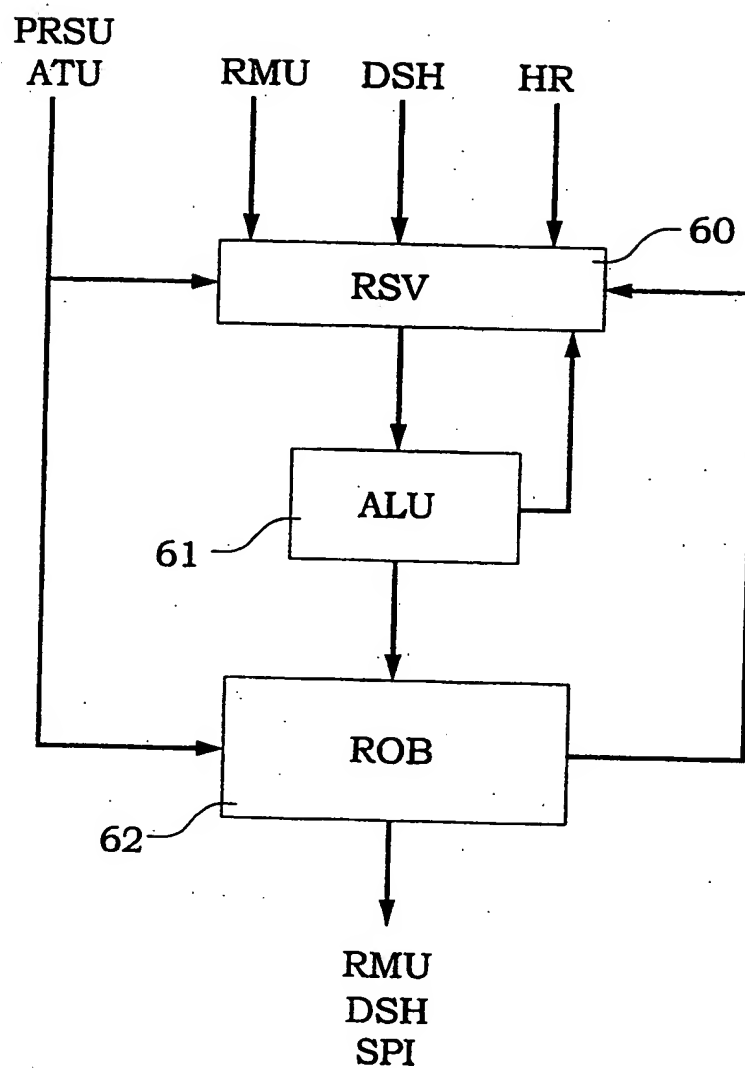
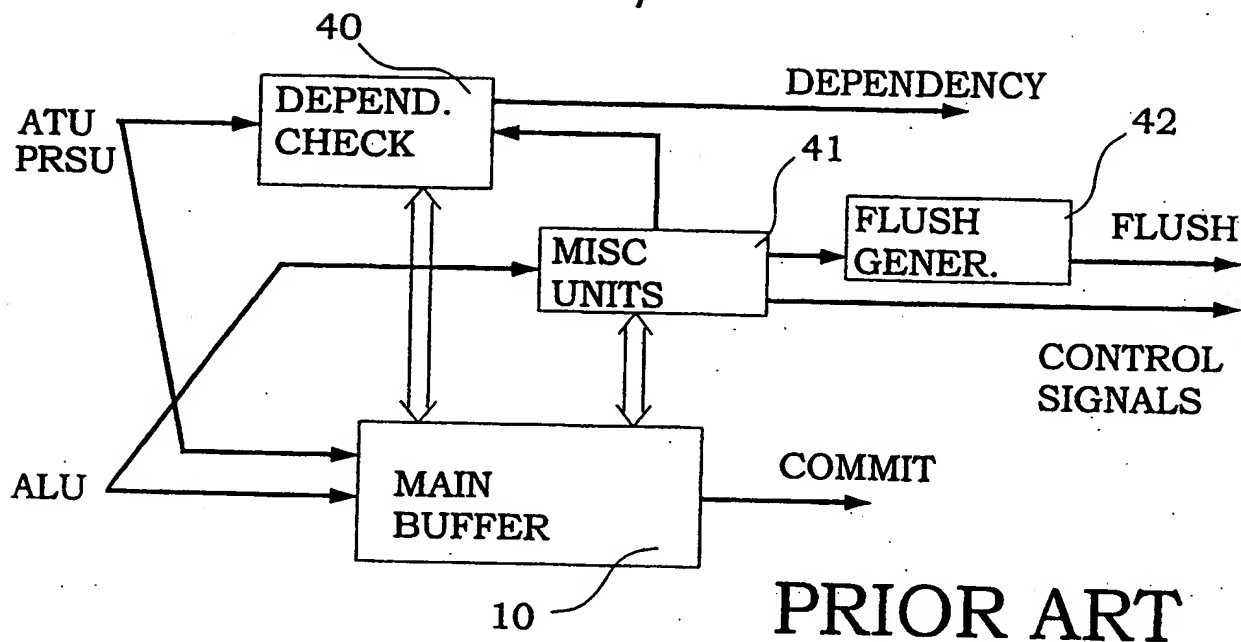


Fig. 3

4/7



PRIOR ART

Fig. 4

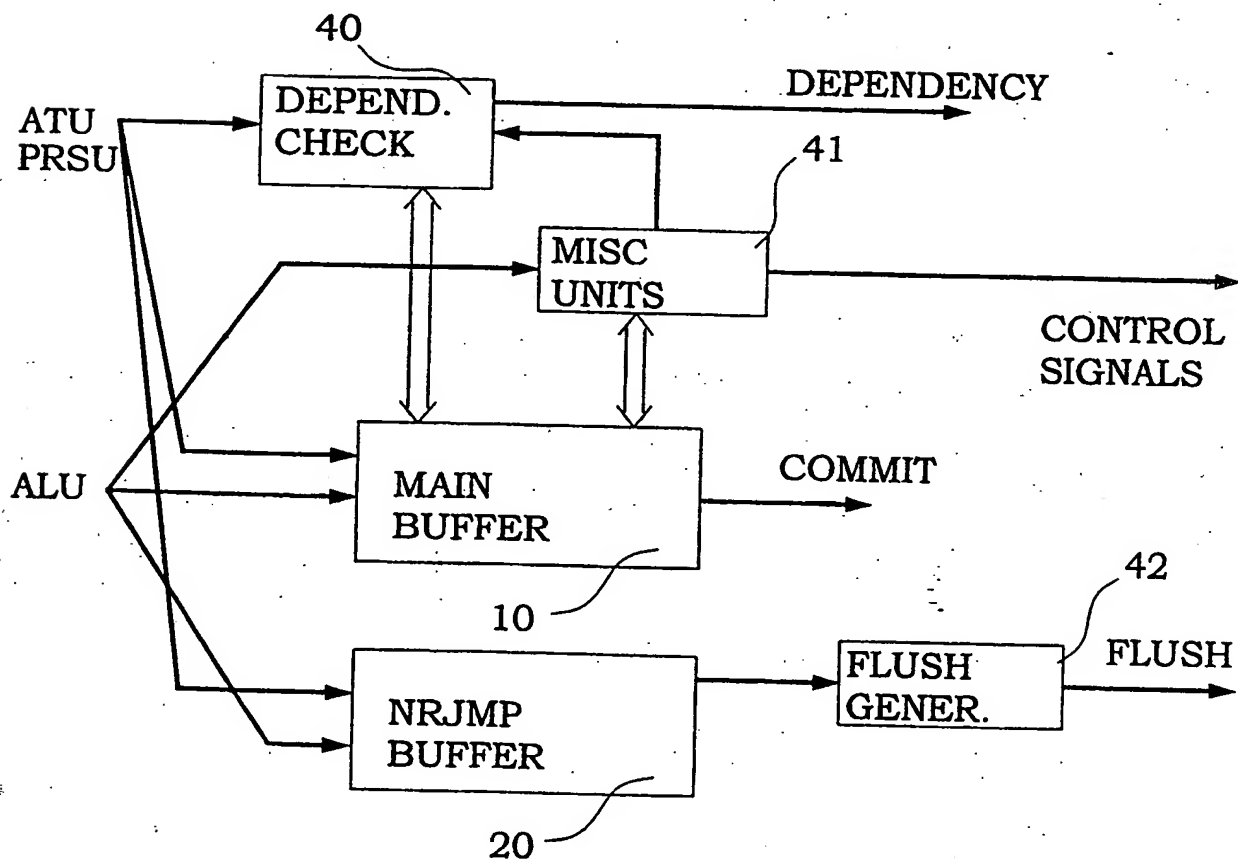


Fig 5

5/7

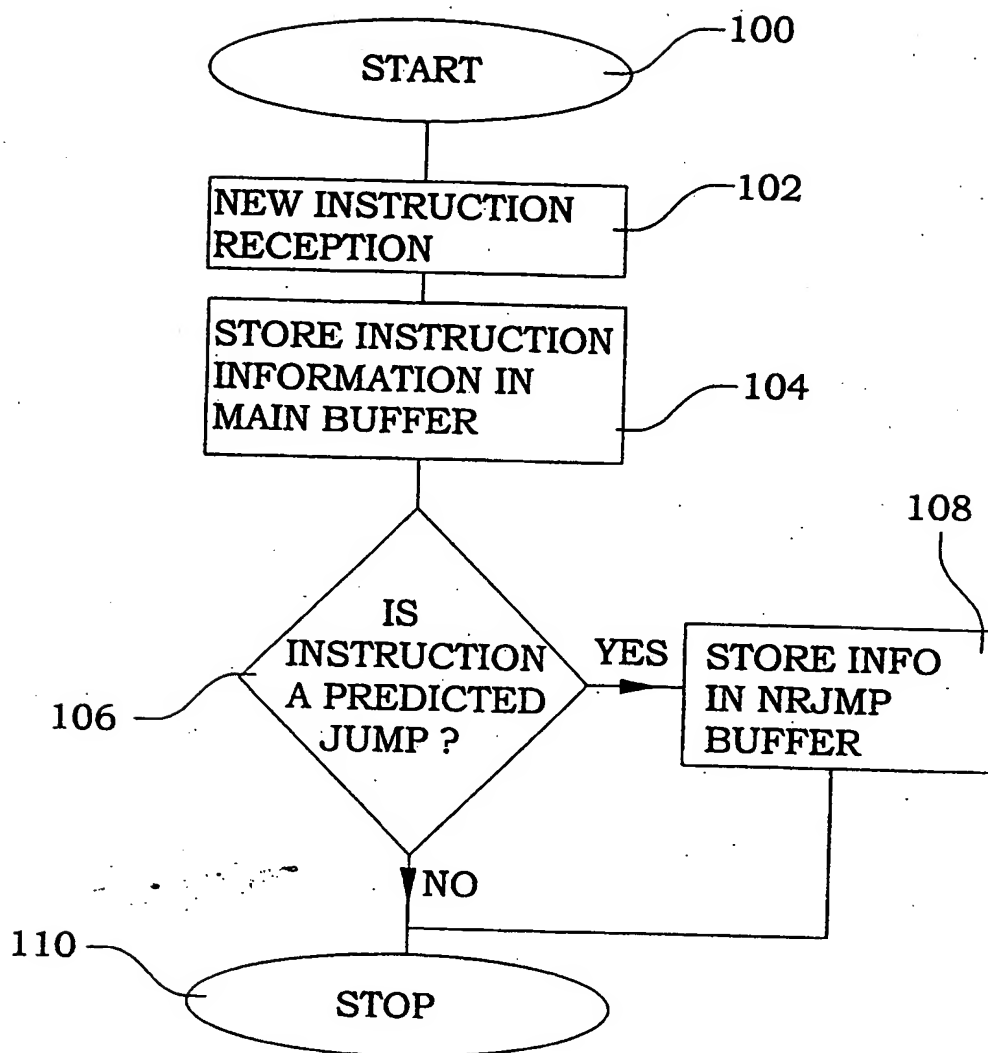


Fig. 6

6/7

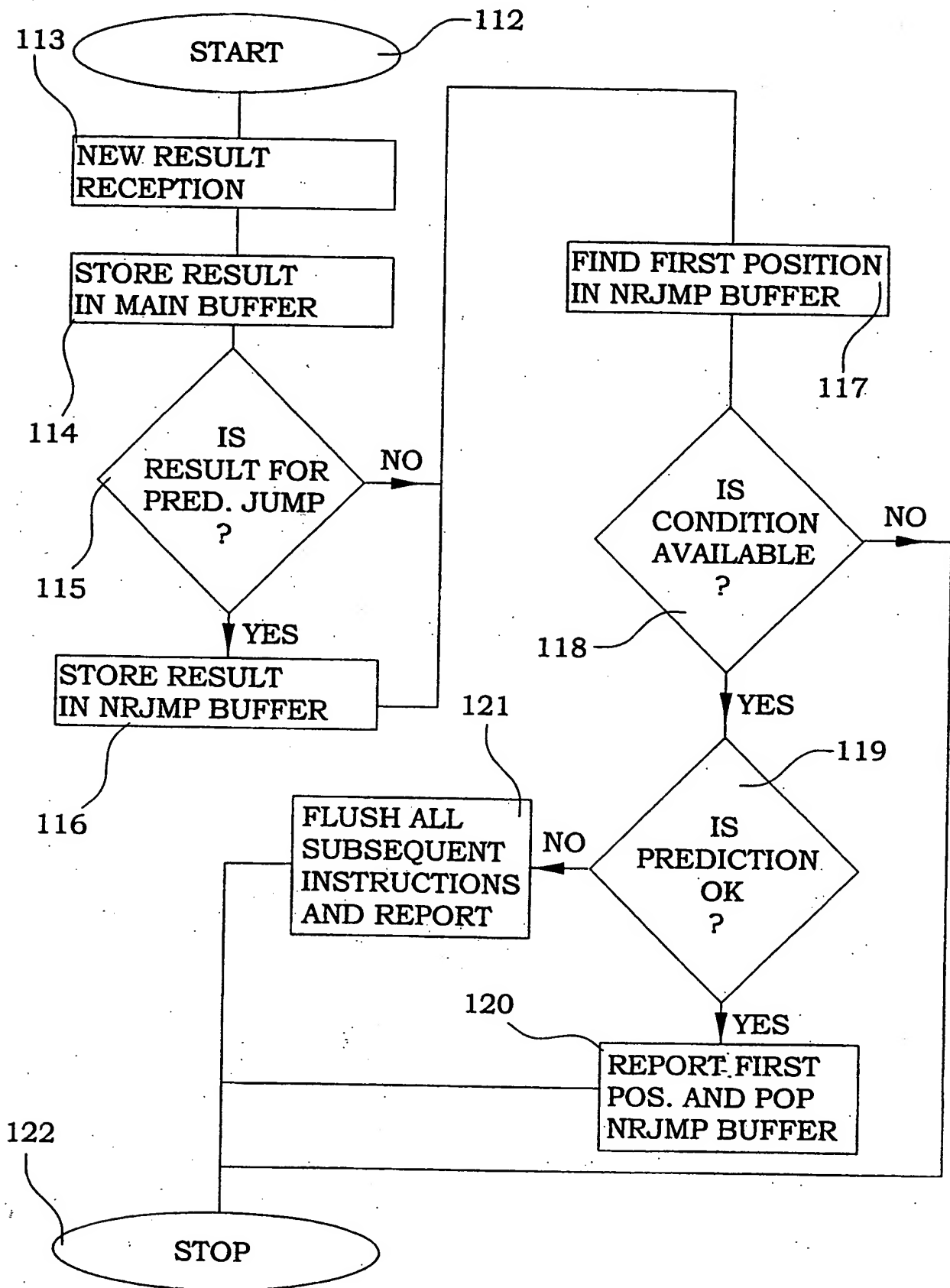


Fig 7

7/7

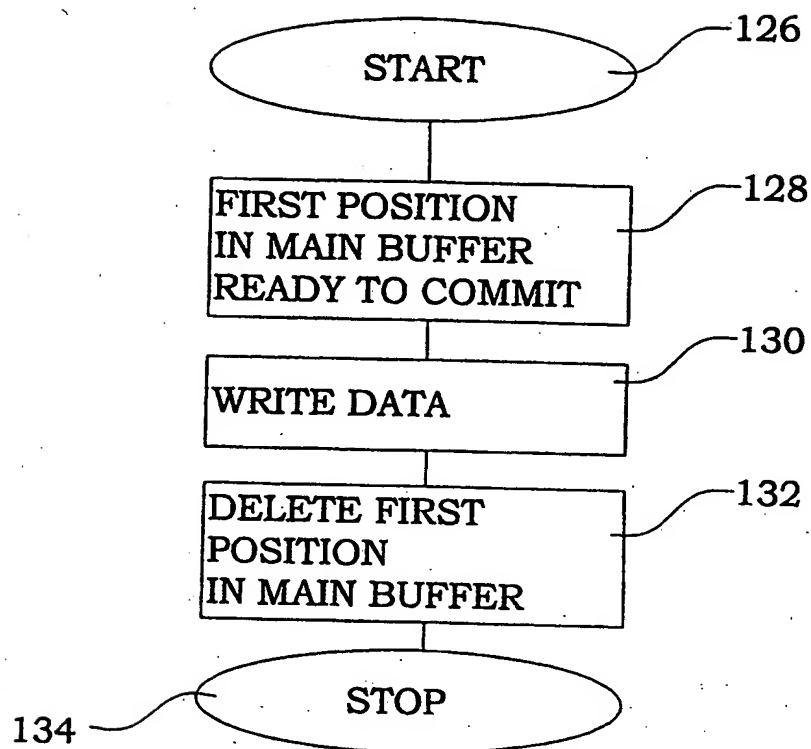


Fig. 8

INTERNATIONAL SEARCH REPORT

International application No.

PCT/SE 99/02361

A. CLASSIFICATION OF SUBJECT MATTER

IPC7: G06F 9/38

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC7: G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

SE,DK,FI,NO classes as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0731408 A2 (INTERNATIONAL COMPUTERS LIMITED), 11 Sept 1996 (11.09.96), whole document --	1-15
A	US 5765016 A (WADE A. WALKER), 9 June 1998 (09.06.98), whole document --	1-15
A	US 5121473 A (STEVEN E. HODGES), 9 June 1992 (09.06.92), column 3, line 29 - column 4, line 14, figure 2 --	1-15
A	US 5778245 A (DAVID B. PAPWORTH ET AL), 7 July 1998 (07.07.98), whole document --	1-15

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"B" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"I" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance: the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance: the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

6 June 2000

Date of mailing of the international search report

14-06-2000

Name and mailing address of the ISA/

Swedish Patent Office

Box 5055, S-102 42 STOCKHOLM

Facsimile No. +46 8 666 02 86

Authorized officer

OSKAR PIHLGREN/EE

INTERNATIONAL SEARCH REPORT

International application No.

PCT/SE 99/02361

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	<p>EP 685789 A2 (ADVANCED MICRO DEVICES INC.), 6 December 1995 (06.12.95), whole document</p> <p style="text-align: center;">-- -----</p>	1-15

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/SE 99/02361

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0731408 A2	11/09/96	AU 700454 B AU 4791796 A DE 69601742 D,T GB 9504743 D US 5721894 A ZA 9601534 A	07/01/99 19/09/96 14/10/99 00/00/00 24/02/98 27/08/96
US 5765016 A	09/06/98	NONE	
US 5121473 A	09/06/92	AU 2654388 A DE 3851746 D,T EP 0320098 A,B	08/06/89 11/05/95 14/06/89
US 5778245 A	07/07/98	NONE	
EP 685789 A2	06/12/95	NONE	